Introduction à la notion d'anticipation et de robustesse dans les problèmes de dial-a-ride dynamiques

Samuel Deleplanque, Alain Quilliot

Introduction à la notion d'anticipation et de robustesse dans les problèmes de dial-a-ride dynamiques

Samuel Deleplanque [1], **Alain Quilliot** [1]

LIMOS, UMR CNRS 6158
Labex IMoBS3
Université Blaise Pascal
Clermont-Ferrand, FRANCE
deleplan@isima.fr

RÉSUMÉ. *Le dial-a-ride problem, noté DARP, est un problème d'optimisation combinatoire associé aux transports à la demande. Il consiste à construire des tournées de véhicules satisfaisant plusieurs requêtes de transport de personnes, ces requêtes entraînant notamment des contraintes de temps difficiles à résoudre. L'expérience montre que les méthodes d'insertion de demandes permettent de fournir au plus vite un rendez-vous aux usagers. L'objet de la recherche menée ici est de proposer une heuristique à base d'insertions successives intégrant une certaine robustesse, obtenue par anticipation des demandes futures, en gardant les tournées flexibles.Nous proposons donc une méthode pour procéder au calcul de l'Insérabilité, à sa prise en compte au moment de la sélection de la demande courante, de la tournée et de sa position au sein de celle-ci, et, enfin, à son utilisation pour aiguiller le calcul des rendez-vous.*

ABSTRACT. *The dial-a-ride problem (DARP) a combinatorial optimization problem related to the on demand transport with hard time constraints. Each user provides a specific demand. The main goal is to build vehicle routing and scheduling which satisfies multiple requests of transportation including hard time constraints. This paper deals with the resolution of these problems. The technique used is a greedy insertion algorithm based on time constraint propagation. In the present work, we integrate a new way to measure the impact of each insertion on the other not inserted demands. We propose its calculation, study its behavior, discuss the transition to dynamic context and present a way to make the system more robust.*

MOTS-CLÉS : *transport à la demande, dial-a-ride problem, Insérabilité, anticipation, robustesse*

KEYWORDS: *on demand transport, dial-a-ride problem, inserability, anticipation, robustness*

DOI:10.3166/HSP.-.1-19

Table des matières

1. Introduction

Les DARP modélisent les problèmes de supervision de flottes de véhicules partagés intégrée dans un service de transport à la demande. Ces services, aujourd'hui proposés aux personnes à mobilité réduite, permettent à leurs usagers de définir leur origine et leur destination en espace et en temps. En effet, en plus de fournir la situation de ces deux points de passage, ils renseignent pour chacun une fenêtre de temps ainsi qu'une durée maximum pour les connecter. L'exploitant requiert également une première contrainte sur la durée maximum de tournée de chaque véhicule et une autre sur leur capacité d'accueil. L'ensemble de ces contraintes forme un des plus difficiles problèmes de tournées de véhicules, il est NP-Difficile. Ce cas a vocation a être utilisé dans un contexte dynamique, les usagers produisant leur requêtes comme bon leur semble, les demandes arrivant donc au fil de l'eau. Dans (S. Deleplanque, 2012) nous proposions une solution heuristique basée sur des techniques d'insertions dont la principale contribution était d'intégrer la propagation des contraintes de temps induites autant par les fenêtres de temps associées à chaque point de passage que par la durée maximum d'acheminement. Partant de cette solution résolvant le DARP statique, nous introduisons dans ce travail un management des flux de demandes intégrant l'évaluation de l'impact de l'insertion d'une demande sur la flexibilité des tournées, l'anticipation des requêtes usagers et leur application dans un contexte dynamique permettant une plus grande robustesse du système. Les exploitations actuelles de transport à la demande sont coûteuses et, de ce fait, rares ; elles nécessitent la plupart du temps des subventions. Pour le rendre plus compétitif, le partage de la capacité d'accueil des véhicules doit être optimisé tout en respectant l'ensemble des contraintes venues des utilisateurs. Il est donc important pour l'exploitant de pouvoir ainsi créer ses tournées tout en prévoyant les demandes probables à venir.

Les sections suivantes se profilent dans cet ordre : suivant un état de l'art sur les problèmes de dial-a-ride, nous présenterons le modèle mais également le framework mis en place dans nos travaux de recherche afin, ensuite, d'introduire la notion d'Insérabilité dans le DARP puis la notion de robustesse reprenant la première notion mais cette fois-ci dans un contexte dynamique. Nous conclurons sur les différentes perspectives ouvertes par ce travail.

2. Etat de l'art sur les problèmes de Dial-a-Ride

Le DARP, dont le *Vehicule Routing Problem* est le premier ancêtre, connait à nouveau un engouement chez les chercheurs après une baisse d'intérêt dans les années 80 pour des raisons technologiques. Cela n'est pas lié à un phénomène de mode mais à un besoin grandissant d'optimiser les coûts, la qualité de service et même aujourd'hui la robustesse. Le DARP connaît également un nombre important de variantes. Alors qu'il a été très étudié en ce qui concerne ses aspects statique ou pseudo-dynamique, peu de travaux sur le DARP ont abouti à une solution efficace et exploitable en temps réel. Le DARP a souvent été l'objet de recherches sur l'optimisation d'un unique critère : la distance totale parcourue par l'ensemble des véhicules. Celui-ci intéresse surtout l'ex-

ploitation et l'empreinte écologique du système. Aujourd'hui, le nombre de critères a augmenté permettant un équilibre entre deux objectifs presque antinomiques : la minimisation du coût et la maximisation du confort des usagers (J. Paquette, 2007). Les problèmes DARP peuvent se décomposer en trois problèmes : un Clustering Problem, un Routing Problem et un Scheduling Problem respectivement un problème d'affectation, un problème de routage et un problème d'horodatage. La résolution de ces trois problèmes peut soit se faire indépendamment ou par un même algorithme (Garaix, 2007). Les méthodes exactes qui ont rencontré le plus de succès pour la résolution du DARP sont à base de programmation dynamique (Psaraftis, 1983) et (R. Chevrier, 2006) ou encore de génération de colonnes souvent utilisées dans la résolution des problèmes de transports (J. Desrosiers, 1995). La complexité du problème qui s'ensuit fait que l'on a tendance à utiliser des méthodes approchées, d'autant plus lorsqu'est envisagée une exploitation réelle (le système doit répondre rapidement aux attentes des usagers). Les heuristiques les plus fréquemment rencontrées sont des variantes d'algorithmes génétiques, du recuit simulé (J. Baugh, 1998), de recherche à grand voisinage (P. Healy, 1995), de la recherche tabou (J.-F. Cordeau, 2003) mais surtout d'algorithmes d'insertions (N. Wilson, 1971), qui ont comme principaux intérêts leur rapidité et leur adaptabilité (J. Jaw, 1986) notamment dans un contexte dynamique comme (O. Madsen, 1995) qui anticipe l'arrivée de nouvelles requêtes en modifiant celles qui sont en cours.

La littérature sur le DARP dynamique est d'ailleurs bien représentée. (A. Colorni, 2001) propose pour le résoudre une solution à base d'insertion, leur mécanisme fait alterner deux phases : la phase de clustering et la phase de routage, cette dernière étant à base de branch-and-cut. (L. Coslovich, 2006) traite encore d'insertions dans un contexte dynamique à propos de routes en train d'être effectuées et tracées suite à l'intégration précédente d'autres demandes. Ils ont établi une mesure de la non satisfaction des usagers et ont fixé un objectif mono-critère : sa minimisation. Les instances traitées considèrent 25 à 50 demandes. Comme ce fut le cas pour le cas statique (J.-F. Cordeau, 2003), la recherche Tabu semble être également une technique efficace de résolution dans un contexte dynamique. (A. Attanasio, 2004), (G. Berbeglia, 2011) puis (Y. Kergosiena, 2011) l'ont mise en place au travers de ce contexte à haute réactivité. Après avoir comparé différentes parallélisations de la recherche Tabu pour le problème DARP statique, (A. Attanasio, 2004) intègre l'implémentation de (J.-F. Cordeau, 2003) dans ce contexte accompagnée d'une procédure *infinite penality* validant les différentes solutions obtenues sur chaque thread. (G. Berbeglia, 2011) reprend encore la même base et l'intègre également au cas dynamique, mais, cette fois-ci, la recherche Tabu s'accompagne d'une procédure exacte de programmation par contrainte. Le troisième, (Y. Kergosiena, 2011), exploite la recherche Tabu dans un contexte réel : le transport de patients. Remarquons au passage que les techniques de résolution du DARP dynamique sont largement utilisées dans de réelles exploitations de transports à la demande, en milieu urbain, de personnes âgées ou handicapées. Mais ce n'est pas tout : en effet, nous pouvons dénombrer bon nombre d'utilisations dans le cadre hospitalier (transports de patients mais aussi de matériel, greffes et médicaments, très contraints en temps). Il est possible de se référer à (A. Beaudry, 2010) qui relève les exemples les plus pertinents. De manière plus générale, (J.F. Cordeau, 2007) fait

l'objet d'un état de l'art poussé sur les solutions au Dial A Ride statique mais aussi dynamique. Malgré une littérature de plus en plus riche dans le domaine, il n'existe pas de modèle de référence notamment sous la forme d'un programme linéaire.

3. Modèle et framework

Il existe de nombreux modèles du DARP mais la plupart ont en commun : une flotte de K véhicules VH de capacité commune CAP, un ensemble de demandes D où chaque demande i, $i = 1..|D|$, est composée de la description de l'origine o_i et la destination d_i (situation 2D et fenêtre de temps $F_{o_i} = [F.min(o_i); F.max(d_i)]$ et F_{d_i}), de la durée maximum qui doit relier ces 2 points et d'une évaluation de la charge à acheminer notée Δ_i. Une solution au DARP est le planning de VH comprenant l'ordre et la date de passage aux origines et aux destinations des demandes, planning qui doit satisfaire l'ensemble des contraintes.

Anticipant un peu sur notre problème, évoquons l'état du système au sein du processus d'insertion des demandes dans les tournées. Le système est composé de :

– un ensemble de demandes déjà traitées $D - D1$ et de demandes à intégrer $D1$,

– l'ensemble des tournées des véhicules de VH noté Γ tel que $\Gamma = \cup_{k \in K} \Gamma_k$ défini par K listes de nœuds bornées par les dépôts et ordonnant l'ordre de passage aux différents nœuds o_i et d_i, $i = 1..|D - D1|$ (nous disposons également d'un état courant des fenêtres de chaque nœud : la propagation de contraintes présentée en section suivante implique une contraction de celles-ci après chaque insertion),

– un ensemble de candidats à l'insertion pour chaque demande de $D1$. Un tel candidat se divise en 5 composants : k le véhicule, i la demande, (x, y) le couple de nœuds de Γ_k (dits points d'ancrage, où respectivement se fait l'insertion de o_i et d_i à droite des nœuds ciblés par ceux-ci) et v la performance après insertion selon les critères envisagés (les problèmes DARP sont en général multi-critères et recherchent un équilibre entre la qualité de service et la minimisation des coûts).

4. Résolution du DARP classique

Le problème classique consiste donc à insérer les demandes de D au sein de l'ensemble des tournées Γ selon une performance établie (minimisation des coûts, maximisation de la satisfaction des clients etc.). La résolution par insertions successives débute par l'initialisation des tournées en insérant dans chacune d'elles deux nœuds dépôts modélisant leur départ et leur arrivée. La boucle principale où chaque itération correspond à l'insertion d'une demande se résume (brièvement) en 4 points :

1. Le processus sélectionne une demande i_0 encore non insérée dans les différentes tournées. Cette sélection se fait selon les contraintes les plus difficiles à satisfaire, soit au sein des demandes où le nombre de véhicules d'accueil possible est petit. Une fois un tel ensemble de demandes sélectionné, on en tire une aléatoirement (si bien sûr, celle-ci n'est pas unique). Si l'heuristique s'exécute dans un processus de

Monte-Carlo, le choix des demandes peut être réalisé selon la difficulté d'insertion lors des itérations précédentes (avec mise en place de pénalités) ;

2. Soit (k_0,x_0,y_0,v_0) un 4-uplet sélectionné tel que k_0 est le véhicule, (x_0,y_0) les deux points d'ancrage et v_0 la performance de VH après insertion. Chacune des demandes possède une liste de tous les 4-uplets possibles. Le choix des modalités d'insertion se fait selon la plus petite différence entre v_0 et l'actuelle performance de VH.

3. i_0 est alors insérée dans la tournée de k_0 avec les nœuds o_{i_0} et d_{i_0} respectivement à droite des nœuds x_0 et y_0 de la liste formant le planning du véhicule k_0.

4. Les calculs suivant l'insertion de i_0 consistent à mettre à jour la liste des 4-uplet de k_0 selon les modifications apportés par i_0. En effet, plus il y a de demandes insérées, plus il y a de points d'ancrage possibles mais à l'inverse moins il y a d'espace-temps au sein des fenêtres pour les futures insertions.

Revenons sur le calcul des 4-uplets qui paramètrent les différentes insertions. La principale difficulté pour les créer concerne la validation des contraintes de temps formées par les demandes, autrement dit, les fenêtres de temps F et la durée maximum d'acheminement Δ.

Le processus teste la non violation des contraintes de temps par propagation lors de l'insertion d'une nouvelle demande dans une tournée. Notons X l'ensemble des noeuds formés par D et $[FP.min(x); FP.max(x)]$ la fenêtre de temps courante de tout nœud x de X, initialisée en entrée par la fenêtre $[F.min(x); F.max(x)]$ fournie par l'usager. Soit R1, R2, R3, R4 et R5 cinq règles d'inférence utilisées dans la propagation de contrainte. R1 et R2 se propagent pour tout couple de nœuds successifs contenant soit un nœud dont la fenêtre a été modifiée par le processus soit un nœud relatif à la demande insérée. R3 et R4 s'appliquent aux couples de nœuds provenant de la même demande (l'un, une origine, l'autre, une destination) et R5 vérifie la validité de l'itération :

– pour tout nœud (x,y) tel que y est successeur de x,

- R1 : $\begin{array}{l} FP.min(x) + DIST(x,y) > FP.min(y) \\ \mid = (FP.min(y) \leftarrow FP.min(x) + DIST(x,y)) \end{array}$,

- R2 : $\begin{array}{l} FP.max(y) - DIST(x,y) < FP.max(x) \\ \mid = (FP.max(x) \leftarrow FP.max(y) - DIST(x,y)) \end{array}$;

– pour tout x et y nœud origine et destination (ou l'inverse) d'une même demande,

- R3 : $\begin{array}{l} FP.min(x) < FP.min(y) - \Delta(x) \\ \mid = (FP.min(x) \leftarrow FP.min(y) - \Delta(x)) \end{array}$,

- R4 : $\begin{array}{l} FP.max(y) > FP.max(x) + \Delta(x) \\ \mid = (FP.max(y) \leftarrow FP.max(x) + \Delta(x)) \end{array}$;

– pour tout x, x$\in \Gamma$,

- R5 : $\begin{array}{l} FP.min(x) > FP.max(x) \\ \mid = \text{REJET} \end{array}$.

Ces 5 règles sont appliquées en boucle tant qu'il existe des fenêtres de temps modifiées à l'itération précédente. Un 4-uplet peut alors être envisagé si ce processus ne renvoie pas REJET. Si t, ensemble des rendez-vous des tournées de VH aux nœuds traversés, a chacun de ses éléments figurant au sein des fenêtres propagées par les règles ci-dessus, les tournées courantes respectent les contraintes de temps. Il ne reste alors à vérifier que la contrainte de charge (la capacité CAP ne doit pas être dépassée) puisque la contrainte sur la durée maximum d'une tournée est testée par la propagation au même titre que les durées Δ, soit par les règles d'inférence R3 et R4 appliquées aux nœuds (indépendants selon les tournées) des dépôts encadrant les listes.

5. Anticipation des futures insertions - Optimisation de l'*Insérabilité*

Dans (S. Deleplanque, 2012), nous avions mis en place une heuristique d'insertions aléatoires des demandes aux performances satisfaisantes résumée par la section précédente. Ces travaux, ainsi que la plupart des propositions de solutions pour la résolution du DARP, prennent peu en compte l'impact de l'insertion d'une demande sur la difficulté future de traitement des demandes restantes à intégrer au sein des tournées. Nous avons vu que les techniques d'insertion sont largement utilisées permettant un passage aisé au contexte dynamique, ces techniques permettant d'être exécutées au sein de tournées partiellement construites. C'est justement dans ce contexte que nous nous positionnons, $D - D1$ étant l'ensemble déjà pris en charge par la flotte VH et $D1$ les demandes restantes à insérer. Nous nous intéressons à l'ordre de sélection des demandes de $D1$ dont l'introduction ne doit pas porter préjudice aux suivantes, quitte, dans certains cas, à en exclure.

Les points d'ancrage renseignent des paramètres d'insertion permettant de garder un bon niveau de performance (QoS et coûts). Nous voulons ici nous concentrer sur le non-rejet d'une demande en considérant que les contraintes les plus à même de bloquer une telle insertion sont formulées par les fenêtres de temps (nous pourrions également prendre en considération la géographie d'une demande ou encore la charge).

Notons INSER(i, Γ) la mesure d'*Insérabilité* (qui peut être considérée comme v) de i sur l'ensemble des tournées Γ et U_n^k l'amplitude courante d'une fenêtre de temps d'un nœud n inséré dans la liste formant le planning du véhicule k. Elle se mesure grâce aux valeurs INSER1 et INSER2 disponibles et mises à jour après chaque insertion. Elles se calculent comme suit :

– $INSER(i, \Gamma) = \sum_{k \in K} INSER1(i, \Gamma(k))$;

– $INSER1(i, \gamma) = Max_{(x,y)} INSER2(i, \gamma, x, y)$ où (x,y) points d'ancrage pour la demande i et γ une tournée de Γ;

– $INSER2(i, \gamma, x, y) = U^{\gamma}_{o_i}(x, y).U^{\gamma}_{d_i}(x, y)$. Les amplitudes sont ici celles résultantes de la propagation des contraintes de temps si l'insertion d est effective.

Nous pouvons dès lors poser le problème de l'optimisation de l'*Insérabilité* en nous positionnant dans l'état courant décrit deux sections plus tôt. Le problème consiste, suite à la sélection de la demande cible i_0, à chercher le véhicule k, $k \in K$ ainsi que le couple de points d'ancrage (x, y) effectuant l'insertion sur la tournée. Pour cela, nous cherchons ces trois paramètres (k, x, y) de telle sorte que $Min_{i \in D-D1}$ $INSER$ $(i, Insertion(\Gamma, i_0, k, x, y))$ ait la plus grande valeur possible avec $Insertion$ (Γ, i_0, k, x, y) la nouvelle collection Γ après insertion de i_0(et contraintes sur les fenêtres de temps propagées).

Remarquons au passage que ce critère de sélection n'a qu'une priorité : celle d'insérer le plus possible de demandes, il est donc nécessaire de combiner ce critère avec la QoS, les coûts etc...

Nous venons de voir comment insérer une demande i_0 tout en maximisant l'Insérabilité des autres demandes de $D1$ par la recherche du triplet (k,x,y) : nous pouvons également nous servir de la plus petite mesure INSER(i, Γ), $\forall i \in D - D1$ pour en déduire i_0.

S'il est possible de rejeter des demandes qui risquent d'avoir de lourdes conséquences sur la possible prise en charge des suivantes, le choix de i_0 doit être modifié pour exclure certaines demandes s'il en résulte une trop forte dégradation de l'*Insérabilité*. Rappelons que si le processus heuristique est un intégré dans un processus de Monte-Carlo, il réalise N itérations du processus principal d'insertions et de propagation. Cela permet de gérer à N reprises une même demande mais dans des conditions différentes d'insertion. Appliquons alors une pénalité à long terme dès lors qu'une demande aura généré une forte dégradation de l'*Insérabilité* des demandes de $D1$.

Une première approche consiste donc à admettre l'idée d'un rejet de i_0 même si celle-ci est insérable. On décidera alors de l'exclure si le nombre de demandes dont l'*Insérabilité* s'est dégradée est excessif. On maximise la quantité $v(Insertion$ $(\Gamma, i_0, k, x, y)) + \mu . \sum_{i \in D1\text{-}\{i_0\}} \Phi(INSER(i, Insertion(\Gamma, i_0, k, x, y)))$, v étant la mesure de qualité standard, μ constante pondérant l'importance du critère d'*Insérabilité* et où Φ est une fonction de R^+ dans [0, 1], croissante, et dont le profil soit de la forme de la fonction $1 - e^{-x}$ avec x de [0,+ ∞], et refuser l'insertion de i_0 dans le cas où la différence de $\sum_{i \in D1\text{-}\{i_0\}} \Phi(INSER(i, Insertion(\Gamma, i_0, k, x, y))$ retranché de $\sum_{i \in D1\text{-}\{i_0\}} \Phi(INSER(i, \Gamma)))$ est supérieure à une certaine quantité que nous appellerons $SEUIL$. Cette quantité pourra être évolutive en cours du processus, et décroître de façon à renforcer les chances d'insérer chaque demande. Dans ce dernier cas, on tentera à nouveau l'insertion des demandes rejetées en fin de processus.

6. Anticipation des futures demandes - tournées robustes

La présente section prolonge naturellement le travail sur l'*Insérabilité* des demandes mais cette fois-ci dans une perspective dynamique. Nous allons mettre en place un processus de résolution qui prend en compte des demandes aléatoires gé-

nérées selon une extrapolation de leur distribution qui pourra être peaufinée au fil de l'eau.

Après avoir précisé la forme prise par la connaissance sur les demandes à venir, il faut l'inclure dans le processus d'insertion. De plus, la gestion du problème en contexte dynamique suppose également, dans la grande majorité des cas, que le traitement des demandes débouche sur des rendez-vous fixés par l'opérateur aux demandeurs. Une fois qu'un tel rendez-vous a été fixé à l'instant t pour le demandeur i, la fenêtre de temps pour o_i voit son amplitude réduite à la seule marge autorisée pour une avance ou un retard sur rendez-vous, soit un petit nombre δ, en général sensiblement inférieur à ce qu'est la fenêtre de temps pour o_i à l'issue du processus d'insertion décrit dans les sections précédentes. Dans la réalité cette marge sur le rendez-vous pourrait par exemple être de 5 minutes alors que la fenêtre de temps, après insertion dans une tournée, était beaucoup plus large. Clairement, la façon dont ces dates de rendez-vous t_{o_i} sont fixées est susceptible d'avoir un impact fort sur l'insertion ou la non insertion des demandes à venir. Il faut donc préciser la façon dont ces dates de rendez-vous vont être calculées.

On notera l'ensemble des demandes virtuelles D-V, D-R celui des demandes réelles et D-$Rejet$ pour les demandes rejetées par le processus d'insertion. D-V sera représenté sous une forme discrétisée, basée sur la notion d'espérance soit D-V=$\sum_{i \in I} p_i i$, où les i sont des demandes classiques et les p_i leur espérance qui modélise le nombre de fois où la demande sera émise durant le futur envisagé. La représentation D-V est susceptible d'impliquer un ensemble I de taille très importante. Or, l'utilisation qui en est faite suppose que I est de taille réduite (au plus une centaine d'éléments). La conséquence en est que la génération de D-V devra induire un processus d'agrégation, tant du réseau à l'intérieur duquel les véhicules circulent que de l'espace temps, de façon à permettre un regroupement des demandes générées par échantillonnage en un nombre limité de demandes.

Comme en fin de section précédente nous pouvons admettre le rejet d'une demande i_0 insérable si l'insertion de celle-ci a des conséquences négatives sur l'insertion des suivantes. Intégrer les demandes futures probables revient à maximiser la quantité donnée en formule 1. En formule 2, le différentiel Π exprime la quantité comparée avec $SEUIL$ impliquant le rejet de la demande si $\Pi < SEUIL$. Les coefficients μ et α sont étalonnés selon l'exploitation.

$$\begin{aligned} & v(Insertion(\Gamma, i_0, k, x, y) \\ & +\mu. \sum_{i \in D1\text{-}\{i_0\}} \Phi(INSER(i, Insertion(\Gamma, i_0, k, x, y)))) \\ & +\alpha \sum_i p_i \Phi(INSER(i, Insertion(\Gamma, i_0, k, x, y))) \end{aligned} \tag{1}$$

$$\begin{aligned}\Pi = \sum_{i \in D1\text{-}\{i_0\}} & (\Phi(INSER(i,\Gamma)) \\ & -\Phi(INSER(i, Insertion(\Gamma, i_0, k, x, y)))) \\ & +(\alpha/\mu).\sum_i p_i.(\Phi(INSER(i,\Gamma)) \\ & -\Phi(INSER(i, Insertion(\Gamma, i_0, k, x, y))))\end{aligned} \tag{2}$$

7. Introduction au problème du rendez-vous

Le DARP consiste à intégrer un ensemble de requêtes de transport à la demande dans un ensemble de tournées Γ. Les paragraphes précédents montraient comment établir un ordre de sélection de ces demandes et/ou sélectionner l'emplacement de celles-ci au sein du plan de route d'un véhicule de la flotte tout en optimisant l'*Insérabilité* des demandes restantes.

L'objet en sortie de ce processus est donc K plans de route intégrant des demandes réelles, un ensemble D-$Rejet$ de demandes exclues par le processus et un ensemble de fenêtres de temps courantes à la fois sur les nœuds des demandes réelles mais aussi sur les virtuelles.

On doit donc à présent fixer aux demandes d acceptées de l'ensemble D-A=D-R−D-$Rejet$, des dates de rendez-vous $t(i)$. De ce fait, fixer ces dates de rendez-vous reviendra, au niveau de la gestion des tournées et des demandes ultérieures, à remplacer les fenêtres $F(o_i)$, i dans D-A, par des fenêtres de temps [$t(i)$,$t(i)$] d'amplitude nulle, ou, si l'on veut introduire un peu de flexibilité via une marge de retard et d'avance, par des fenêtres de temps [$t(i)$ - δ, $t(i)$+ δ], où δ est un nombre relativement petit supérieur à zéro.

Il faut souligner, hors cas de retard, qu'une fois le rendez-vous fixé avec l'usager, ce dernier n'acceptera pas de changement de dernière minute, ce sera la dernière réduction de la fenêtre (jusqu'à une amplitude nulle). Au passage, dans le futur processus d'intégration de nouvelles demandes, le temps nécessaire à la propagation qui devrait s'appliquer sur une telle fenêtre sera évidemment raccourci.

Le critère principal aiguillant le calcul de ces dates de rendez-vous devrait être le même que le critère de performance global, c'est-à-dire qu'il s'agirait de maximiser la qualité des tournées au sens usuel, sans dégrader l'*Insérabilité* des demandes virtuelles de D-V. Plus précisément, on voudra, tout en gardant le fait que les tournées Γ (k), k = 1..K, satisfont les contraintes auxquelles elles sont soumises, maximiser la quantité $v(\Gamma)+\alpha.\sum_i p_i.\Phi(INSER(i,\Gamma))$. Ces quantités étant recalculées sur la base de fenêtres de temps $F(o_i)$, i dans $D-A$, égales à [t(i), t(i)], et d'amplitude nulle. On dispose d'un ensemble X de variables qui sont associées à chacun des points de passage des tournées de Γ, ainsi qu'un deuxième ensemble Y associé aux demandes virtuelles ; on souhaite alors déterminer les valeurs prises par ces variables, de telle

sorte que (entre autres) les fenêtres de temps associées aux variables de Y demeurent les plus grandes possible. Autrement dit, et c'est l'objet principal de cette recherche, le processus doit créer des tournées au fil de l'eau de façon à laisser une capacité d'accueil sur le chemin des véhicules aux demandes communes. Rappelons que nous nous basons sur l'amplitude des fenêtres, et, dans un contexte dynamique, sur l'amplitude des demandes qui sont susceptibles d'apparaître lors des prochains appels de la procédure générale d'insertion. Ici nous positionnons alors la date de rendez-vous de façon à minimiser la contraction des fenêtres de temps de ces demandes virtuelles et ceci de manière heuristique (nécessaire si l'on se réfère à la complexité du problème). Le traitement est réalisé soit exclusivement sur les rendez-vous aux origines des demandes soit, si le type d'exploitation le requiert (ex. un transport multimodal), sur l'ensemble des nœuds traversés par les véhicules (nœuds dépôts et destinations compris). Nous nous reposerons sur le premier cas.

L'heuristique utilisée est relativement simple, elle consiste à balayer l'ensemble des demandes virtuelles et, pour chacune d'elles, à étudier son insertion par les paramètres (k, x, y) qui fournissent la meilleure valeur $INSER2(i, k, x, y)$. Nous traitons alors des points d'ancrage x ou y dont le statut associé au nœud le caractérise comme une origine. Suite à la propagation de contraintes, nous contractons une nouvelle fois les fenêtres de temps afin que, par exemple, pour x point d'ancrage de l'origine virtuelle o_v et dist() fonction retournant le plus court chemin entre deux nœuds :

- F.Min(x) = Sup(F.Min(x), F.Min(o_v + Dist(o_v,x)) ;
- F.Max(x) = Inf(F.Max(x), F.Max(o_v) - Dist(x, o_v)).

On applique ce processus pour toute demande virtuelle associée au couple (k, x, y) le plus performant. Une fois que toutes les fenêtres des nœuds origines des demandes réelles insérées sont ainsi contractées, on place les rendez-vous au sein de ces fenêtres de façon à optimiser la performance de la flotte (comme dans (S. Deleplanque, 2012)) selon les critères choisis (minimisation des distances et/ou minimisation des durées des tournées etc...).

8. Le DARP dynamique

La qualité de la solution fournie par le processus présenté dans ce travail ne peut être établie seulement dans le cadre d'une simulation. Le DARP dynamique se caractérise par le fait que les demandes arrivent au fil de l'eau et que les décisions de routage doivent être prises en conséquence. Formellement, un algorithme de gestion dynamique considère, à un instant donné t, un état de l'ensemble des tournées du système défini par les routes en cours, un état des demandes non traitées, une représentation probabiliste des demandes à venir D-V, et décide, soit de ne rien faire, soit de traiter certaines demandes de D et de rejeter les autres. Le traitement des demandes acceptées implique l'affectation de ces demandes sur le planning des véhicules, ainsi que la détermination des dates de rendez-vous décrite plus haut. La simulation scinde la spécification d'un tel algorithme en deux processus : DEC et ROUTE.

Le premier, DEC déclenche ou non le processus de traitement des demandes en cours, autrement dit le processus présenté dans ce papier. Les éléments d'Input qui vont déterminer le résultat de DEC seront principalement les suivants :

– Le nombre de demandes en attente ;

– L'urgence de ces demandes, c'est-à-dire, pour chaque demande i, l'écart entre l'instant courant et le milieu de la fenêtre associée à o_i.

Le second, appelé ROUTE, gère les demandes en cours. L'Input qui lui est associé est principalement celui du DARP dans sa forme classique, soit :

– Le réseau X complet comprenant les nœuds affectés aux véhicules mais qui ne sont pas encore desservis au moment du déclenchement de ROUTE. Les nœuds dont la fonction est l'origine d'une demande, se voient accompagnés d'une fenêtre de temps réduite [t(x) - δ, t(x) + δ] où t(x) est une date de rendez-vous et δ une courte marge de retard/avance pouvant être nulle.

– Les tournées des K véhicules de Γ construites sur les nœuds de X. Le premier nœud de la liste formant le planning d'un véhicule correspond à la position courante du véhicule.

– L'ensemble des demandes non encore traitées.

– L'ensemble des demandes aléatoires sous la forme $\sum_{i \in I} p_i.i$ où les demandes i sont des demandes définies sur un réseau réel réduit : elles représentent les demandes à venir sur un intervalle de temps significatif pour une bonne prise en compte des demandes.

9. Expérimentations

9.1. L'heuristique à base de propagation de contraintes

(Cordeau, 2006) utilise un algorithme de Branch-and-cut afin de résoudre leurs propres instances où la performance est mesurée sur un seul critère : la minimisation des distances parcourues. Pour nos premières expériences, nous les avons reprises et résolues afin de pouvoir tester l'efficacité de notre algorithme à base de propagation de contraintes. 12 instances sont résolues où les demandes se positionnent au sein d'un carré de coté 20 (il faut 20 minutes pour le traverser), la capacité des véhicules est de 3 et les durées maximum d'acheminement de 30 (minutes). Le nom des instances indique K et |D|, il est de la forme aK-|D|. La tableau 1 reprend les meilleurs résultats obtenus de (Cordeau, 2006) et (Parragh, 2011) (lb = lower bound, cpu* = CPU littérature en secondes, Opt l'optimal si connu, TI notre résultat, Gap en pourcentage entre TI et Opt et cpu nos temps CPU (également en secondes) pour atteindre la meilleure solution).

D'emblée, nous pouvons nous satisfaire de retrouver très souvent les solutions optimales fournies par la littérature ou sinon essayer de nous en approcher très fortement, le plus gros gap étant de 1,48%. De plus, les temps pour les obtenir sont, dans la plu-

Tableau 1. Gap et temps CPU (secondes)

$Inst.$	Lb	Opt	$cpu*$	TI	$Gap\%$	cpu
$a2-16$	$294,25$	$294,25$	1	$294,25$	$0,00$	0
$a2-20$	$344,83$	$344,83$	3	$344,83$	$0,00$	0
$a2-24$	$431,12$	$431,12$	9	$431,12$	$0,00$	0
$a3-18$	$300,48$	$300,48$	5	$300,81$	$0,11$	1
$a3-24$	$344,83$	$344,83$	8	$344,83$	$0,00$	2
$a3-30$	$494,85$	$494,85$	10	$495,26$	$0,08$	16
$a3-36$	$583,19$	$583,19$	105	$589,86$	$1,14$	14
$a4-16$	$282,68$	$282,68$	6	$283,10$	$0,15$	0
$a4-24$	$375,02$	$375,02$	6	$376,21$	$0,32$	94
$a4-32$	$485,50$	$485,50$	31	$487,10$	$0,33$	29
$a4-40$	$557,69$	$557,69$	8328	$565,95$	$1,48$	63
$a4-48$	$668,82$	NA	14543	$700,30$	NA	31

part des cas, bien inférieurs aux travaux de recherches cités. Il faut surtout remarquer que les temps n'explosent pas comme c'est le cas dans la littérature pour les deux dernières instances. Enfin, nous pouvons conclure qu'il est tout à fait possible d'intégrer la minimisation de la distance parcourue dans le calcul de v comme un objectif compatible avec notre solution.

9.2. Etude du comportement de la mesure d'Insérabilité

Nous avons relevé les différentes valeurs de la mesure d'*Insérabilité INSER* lors d'un échec d'une réplication du processus de résolution sur l'instance la plus difficile de (J.-F. Cordeau, 2003) : la pr10. La difficulté de ces instances est principalement concentrée sur les fenêtres de temps. Nous étudions donc ici une réplication "sans succès" où l'*Insérabilité* tend vers 0. Notre heuristique à base propagation de contraintes utilise la mesure d"*Insérabilité* au moment de la sélection de la demande à intégrer : systématiquement, c'est la demande à la plus base valeur *INSER* qui sera insérée.

Plus de 4500 valeurs ont été relevées au fil du processus tentant d'insérer les 144 demandes. Les valeurs correspondent bien entendu aux demandes restantes à insérer et leur propre mesure d'*Insérabilité* varie au fil des insertions des autres demandes. Voir le graphique de la figure 1 d'ordonnée *INSER* formée par l'amplitude des fenêtres contractées par la passage des règles d'inférences et d'abscisse le numéro du relevé qui peut se comparer au fil d'exécution.

En début de processus, le nombre de demandes à insérer est important d'où le grand éventail de valeur pour *INSER* (de 15000 à 40000). Au fait de l'exécution, et notamment entre les relevés 2000 et 3000, les *INSER* subissent de fortes baisses pour certaines demandes qui sont pour le coup sélectionnées et insérées (d'où une remontée des plus basses valeurs à partir du relevé 3000). Aussi, en fin de processus, les demandes restantes subissent une forte dégradation de leur propre *INSER* suite aux

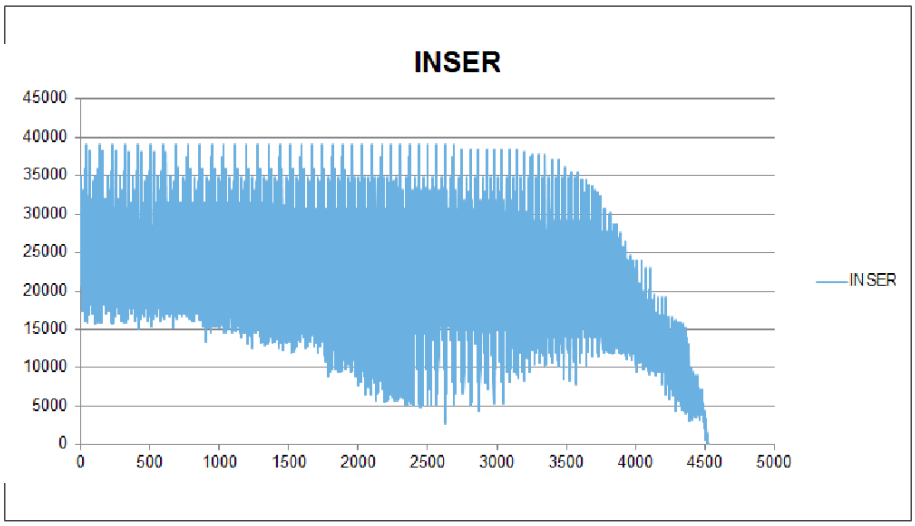

Figure 1. Variation de l'Insérabilité

insertions déjà réalisées qui ne laissent plus assez de place pour l'ensemble. Nous obtenons donc naturellement une valeur nulle en fin de cette réplication qui a échoué.

9.3. Insérabilité dans la sélection des paramètres d'insertion - Optimisation de l'Insérabilité et exclusion de demandes

Les instances traitées ci-dessus ont été réalisées de façon à ce que toutes les demandes puissent être insérées. Nous nous préoccupons maintenant de l'étude de la mesure *d'Insérabilité* dans un autre contexte : il s'agit d'insérer le plus possible de demandes sans qu'il soit possible pour autant de le faire pour toutes. Nous allons donc générer de nouvelles instances avec, cette fois-ci, l'objectif d'obtenir le plus grand pourcentage de demande insérées. Nous comparerons ici trois heuristiques :

1. l'heuristique d'insertions aléatoires à base propagation de contraintes où seule la minimisation des distances est recherchée ;
2. une seconde heuristique d'insertions aléatoires basée sur la résolution du problème de l'optimisation basé sur la maximisation de $\sum_{i \in D1 - i_r} INSER(i, Inserted(\Gamma, i_r, k, x, y))$;
3. une dernière heuristique identique à la précedente, en y ajoutant la possibilité d'exclure certaines demandes dont la prise en charge implique une trop grande variation de l'*Insérabilité* des demandes restantes.

Ici, la sélection de la demande selon la plus basse valeur INSER n'a pas lieu d'être utilisée. D étant trop important, le pourcentage de demandes insérées serait moindre puisque le processus tenterait d'insérer des demandes dont l'insertion est la plus difficile et ceci pour deux raisons : les contraintes de temps ou alors la *marginalité* de la

demande qui force les véhicules à s'écarter largement de leur planning intial. Comme évoqué début d'étude, les demandes marginales pourraient être pondérées selon la différence *gain - coût* afin de déterminer si l'exploitation qui est faite de ces algorithmes doit inclure ou pas ce type de demandes. Ces trois heuristiques se baseront alors sur les plus petites cardinalités de *LibreCar* de chaque demande.

En comparant ces heuristiques, nous allons étudier l'utilité de la mesure d'*Insérabilité* dans la sélection des paramètres *(x,y,k)* et de l'exclusion de demandes dont l'insertion pénaliserait les prochaines.

9.3.1. Sensibilité de l'Insérabilité selon des composants de l'Input

La génération d'instances n'est jamais une étape évidente. En effet, la prise en compte de la mesure d'*Insérabilité* ne trouve pas toujours son utilité. Plaçons-nous dans un contexte statique où toutes les données sont connues en amont de la réalisation et où tout ensemble de demande peut être facilement inséré, cette mesure n'a évidemment pas lieu d'être. Son intérêt dépend également du choix d'insertion. Plus la cardinalité des ensembles *LibreXY* est importante, plus il y a différentes manières d'insérer un demande donnée et plus il y a de manières de sacrifier l'insertion des futures demandes. Il faut également avoir en tête que, dans un cas où *D* ne pourra être intégrée dans son entier dans Γ, le nombre de demandes et leur homogénéïté est un élément clé dont la variation détermine le besoin d'une mesure de l'*Insérabilité*. Autrement dit, la facilité de résolution d'une instance ayant un ensemble *D* avec trop peu de demandes ne requiert pas l'utilisation de cette mesure tout comme un ensemble *D* avec un nombre très important de demandes (et avec l'optimisation de la valeur $Min_{i \in D1-i_r} INSER(i, Inserted(\Gamma, i_r, k, x, y))$.

Pour illustrer ces quelques remarques sur le volume de demandes, fixons les paramètres d'entrée par les données du tableau 2 avec $e_{F(o)}$ et $e_{F(d)}$ respectivement les amplitudes des fenêtres de temps à l'origine et à la destination homogènes sur *D*. Selon 5 volumes de demandes chacun comptabilisant 5 instances, le tableau 3 relève les T_{Insert} et $T_{Insert_{Rob}}$ des taux d'insertion alors obtenus respectivement avec 10 réplications des deux premierse heuristiques.

Tableau 2. Paramètres de génération d'instances

K	$e_{F(o)}$	$e_{F(d)}$	Δ	CAP
10	35	10	∞	10

Tableau 3. Gap entre les taux INSER

$\|D\|$	50	75	100	150	200
T_{Insert}	100	93.2	78.9	64.2	52.6
$T_{Insert_{Rob}}$	100	96.8	85.3	66.4	54.1
Gap_{Insert}	0	3.86	8.11	3.43	2.81

Nous voyons donc ici l'importance d'étudier les données d'un problème avant d'établir la nécessité d'intégrer une mesure d'*I*nsérabilité. Remarquons au passage

que, lorsque c'est nécessaire, nous obtenons une augmentation qui peut aller jusqu'à 8% de demandes insérées en plus grâce à la sélection des paramètres *(x,y,k)* selon la variation de l'*Insérabilité*.

9.3.2. Demandes exclues : étude du seuil

Cette courte section étudie l'exclusion d'une demande lorsqu'elle est décidée suite à une trop grande variation de l'*Insérabilité* des demandes restantes si la première avait été intégrée à Γ . Nous avons vu que cette variation était calculée par la somme $\sum_{i \in D1\text{-}\{i_0\}}(\Phi(INSER(i,\Gamma))) - \Phi(INSER(i, Inserted(\Gamma, i_0, k, x, y)))$ si i_0 est la demande sélectionnée et (x,y,k) ses paramètres d'insertion. Passée une valeur *SEUIL*, cette différence induit le rejet de la demande même si les paramètres *(x,y,k)* ont été optimisés au préalable. La difficulté est d'établir cette valeur *SEUIL*. Prenons la résolution d'une instance générée selon les paramètres donnés un peu plus haut et relevons toutes les variations de l'*Insérabilité* une fois (i_o, x, y, k) sélectionnés. La figure 2 relève 19 de des variations tout au long de l'heuristique avec *INSERav* et *INSERap* respectivement les valeurs $\sum_{i \in D1\text{-}\{i_0\}} \Phi(INSER(i,\Gamma))$ et $\sum_{i \in D1\text{-}\{i_0\}} \Phi(INSER\ (i, Inserted(\Gamma, i_0, k, x, y)))$.

Un simple coup d'oeil au diagramme de la figure 2 nous amène à remarquer qu'il est nécessaire d'avoir une valeur *SEUIL* dynamique dans le sens où elle doit décroître au fil de l'exécution. Sa décroissance se basera sur un taux relatif à $\sum_{i \in D1\text{-}\{i_0\}}(\Phi(INSER(i,\Gamma)))$.

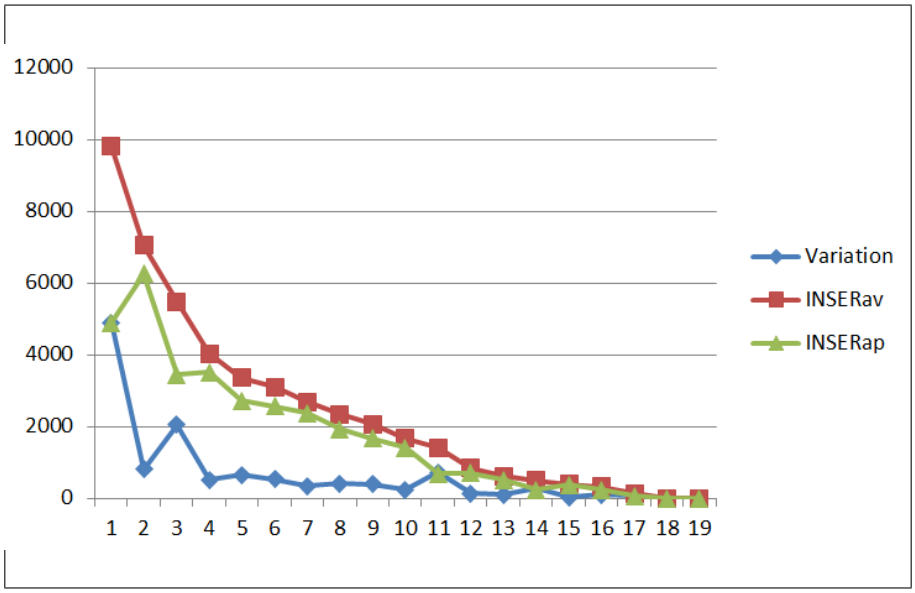

Figure 2. Variation des valeurs $INSER$

Il est nécessaire de souligner à nouveau que cette technique peut exclure systématiquement les demandes les plus marginales qui pourraient par exemple traverser tout

l'espace dans lequel les véhicules évoluent. Pour l'instant, nous permettons à notre programme de tenter à nouveau l'insertion des demandes exclues en fin de processus mais, ceci peut ne pas être suffisant si les tournées sont devenues très peu flexibles.

Nous reprenons maintenant la série d'instances pour laquelle la mesure d'Insérabilité trouvait tout son interêt (100 demandes pour 10 véhicules). Nous appliquons notre troisième heuristique qui exclut une demande si son insertion dégrade trop fortement la probable insertion des éléments de $D1 - i_0$. Nous obtenons alors un taux moyen $T_{Insert_{Rob}}$ de 86,6% au lieu des 85,3%.

L'intérêt semble moins évident que la simple résolution du problème de l'Insérabilité mais il faut souligner que ce rejet dépend de toutes les demandes restantes comprenant alors celles qui pourront également être exclues à leur tour.

10. Perspectives et conclusion

Les traitements du DARP par insertions successives des demandes, tels qu'ils ont pu être réalisés dans la littérature, souffrent d'un défaut : lorsqu'une demande est insérée dans un ensemble de tournées courantes, il est peu tenu compte de l'impact de cette insertion sur la difficulté qu'il pourra y avoir à traiter les demandes restant à insérer. Le but de cette recherche était donc d'aborder ce problème, et de montrer de quelle manière il est possible de prendre en compte les insertions à venir. Nous avons mis en place un calcul de l'*Insérabilité* des demandes puis son implication aux moments de la sélection d'une demande, du choix de son emplacement dans une tournée et enfin du calcul des rendez-vous.

Nos travaux intègrent une notion de robustesse envisageable dans bon nombre de variantes des dial-a-ride problems. Dans nos expériences, le problème de fixation des rendez-vous ne s'appliquait que sur les origines des demandes, nous pouvons envisager de l'exploiter au sein du Dial-a-ride avec transbordement ou division du chargement (S. Deleplanque, 2013a) et (S. Deleplanque, 2013b), où il faudra élargir ces nœuds aux nœuds relais ou encore évaluer l'*Insérabilité* lorsqu'une demande est prise en charge par plusieurs véhicules (soit par division d'une demande d'acheminement d'un couple personnes/marchandises ou par le transfert de ce chargement d'un véhicule à un autre).

Bibliographie

A. Attanasio G. G. G. L., J.F. Cordeau. (2004). Parallel tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. *Parallel Computing. Volume 30, Issue 3, Page 377-387*.

A. Beaudry T. M. S. N., G Laporte. (2010). Dynamic transportation of patients in hospitals. *OR Spectrum, Volume 32, Issue 1, 77-107*.

A. Colorni G. R. (2001). Modeling and optimizing dynamic dial-a-ride problems. *International Transactions in Operational Research - Volume 8, Issue 2, pages 155–166, March 2001*.

Cordeau J.-F. (2006). A branch-and-cut algorithm for the dial-a-ride. *Operation Research. Vol, 54, n°3, p 573-586*.

Garaix T. (2007). Etude de résolution exacte de problèmes de transport à la demande avec qualité de service. *Thèse de doctorat. Soutenue en décembre 2007*.

G. Berbeglia G. L., JF Cordeau. (2011). A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem. *JOC - INFORMS Journal on Computing*.

J. Baugh R. K. J. S., G. Krishna. (1998). Intractability of the dial-a-ride problem and a multiobjective solution using simulated annealing. *Engineering Optimization 30, 91–123*.

J. Desrosiers M. S. F. S., Y. Dumas. (1995). Time constrained routing and scheduling. *Volume 8 dans Handbooks in Operations Research and Management Science, 35–139. Amsterdam, North-Holland : Informs*.

J.-F. Cordeau G. L. (2003). A tabu search heuristic algorithm for the static multi-vehicle dial-a-ride problem. *Transportation Research B 37, 579–594*.

J.F. Cordeau G. L. (2007). The dial-a-ride problem: models and algorithms. *Annals of Operations Research*.

J. Jaw H. P. N. W., A. Odoni. (1986). A heuristic algorithm for the multi-vehicle many-to-many advance request dial-a-ride problem. *Transportation Research B 20B, 243–257*.

J. Paquette G. L., J.F. Cordeau. (2007). Etude comparative de divers modèles pour le problème de transport à la demande. *Rapport technique. CIRRELT, HEC Montréal*.

L. Coslovich W. U., R. Pesenti. (2006). A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. *European Journal of Operational Research 175, 1605-1615*.

N. Wilson H. W. B. H., J. Sussman. (1971). Scheduling algorithms for dial-a-ride systems. *Rapport technique, Urban Systems Laboratory,Massachusetts Institute of Technology, Cambridge, MA*.

O. Madsen J. R., H. Ravn. (1995). A heuristic algorithm for the a dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of Operations Research 60, 193–208*.

Parragh S. (2011). Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. *Transportation Research Part C: Emerging Technologies*.

P. Healy R. M. (1995). A new extension of local search applied to the dial-a-ride problem. *European Journal of Operational Research 83, 83–104*.

Psaraftis H. (1983). An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows. *Transportation Science 17, 351–357*.

R. Chevrier P. C. D. J., P. Canalda. (2006). Comparison of three algorithms for solving the convergent demand responsive transportation problem. *ITSC'2006, 9th Int. IEEE Conf. on IntelligentTransportation Systems, Toronto, Canada, 1096–1101*.

S. Deleplanque A. Q. (2012). Insertion techniques and constraint propagation for the darp. *Computer Science and Information Systems (FedCSIS), IEEE Conference publications, 393-400*.

S. Deleplanque A. Q. (2013a). Constraint propagation for the dial-a-ride problem with split loads. *Recent Advances in Computational Optimization. Studies in Computational Intelligence, Vol. 470. ISBN 978-3-319-00409-9,Springer, Volume 470, pp 31-50*.

S. Deleplanque A. Q. (2013b). Transfers in the on-demand transportation: the dial-a-ride problem with transfers. *MISTA 2013 - Ghent, Belgium, (ISSN 2305-249X),(Parution 2013, 20 pages)*.

Y. Kergosiena D. P. J. B., Ch. Lentéa. (2011). A tabu search heuristic for the dynamic transportation of patients between care units. *European Journal of Operational Research Volume 214, Issue 2, 16 October 2011, Pages 442–452*.